# Digital Image Processing

Image Restoration:

Noise Removal

# Contents

In this lecture we will look at image restoration techniques used for noise removal

- What is image restoration?
- Noise and images
- Noise models
- Noise removal using spatial domain filtering
- Periodic noise
- Noise removal using frequency domain filtering

# Image restoration

- Image restoration is the process of recovering the original scene from the observed scene which is degraded.

- Different form enhancement- aim of enhancement techniques make images visually appealing, whereas restoration essentially inverts the degradation- more objective.

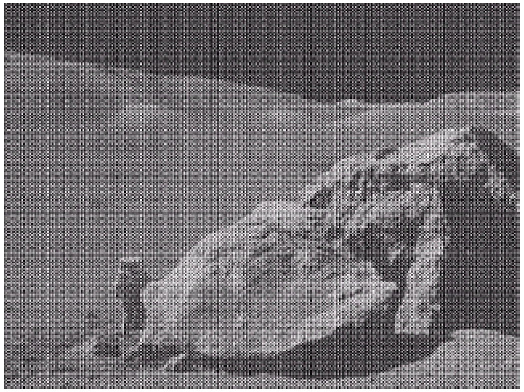
# Image restoration

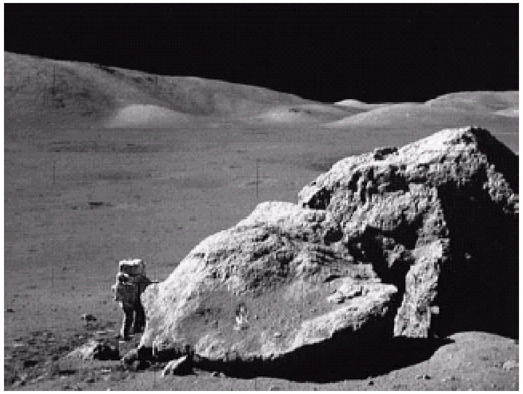
- Has intersections with signal processing, estimation theory, inverse problems, linear algebra.

- As with most of the topics in Image Processing, restoration was developed to be used in astronomy, and is also used for restoring old films and pictures

# What is Image Restoration?

- Image restoration attempts to restore images that have been degraded
- Identify the degradation process and attempt to reverse it
  - Similar to image enhancement, but more objective





# Noise and Images

The sources of noise in digital images arise during image acquisition (digitization) and transmission

- Imaging sensors can be affected by ambient conditions
- Interference can be added to an image during transmission

# Noise Model

We can consider a noisy image to be modelled as follows:

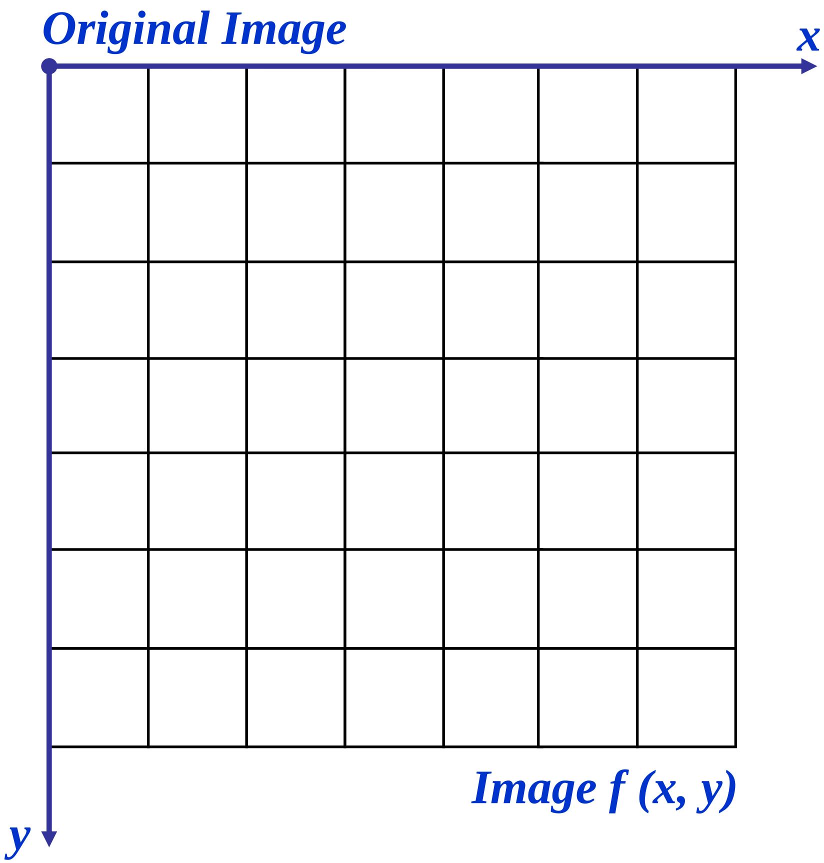
$$g(x, y) = f(x, y) + \eta(x, y)$$

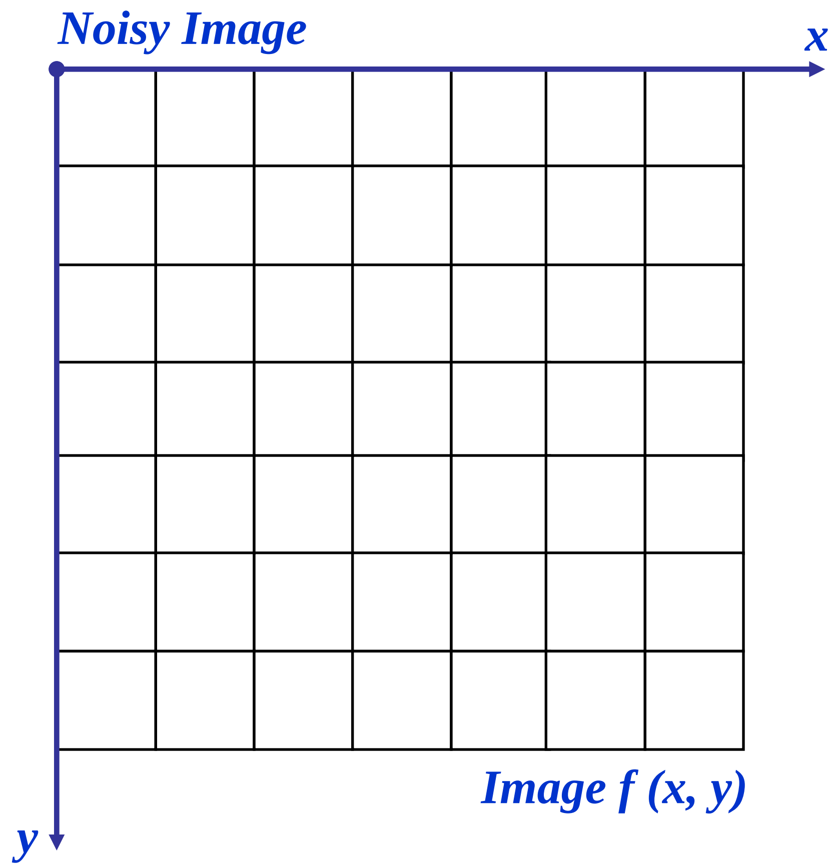
where $f(x, y)$ is the original image pixel, $\eta(x, y)$ is the noise term and $g(x, y)$ is the resulting noisy pixel

- If we can estimate the model of the noise in an image, this will help us to figure out how to restore the image

- Noise is assumed to be uncorrelated with pixel values and does not depend on spatial coordinates

# Noise Corruption Example



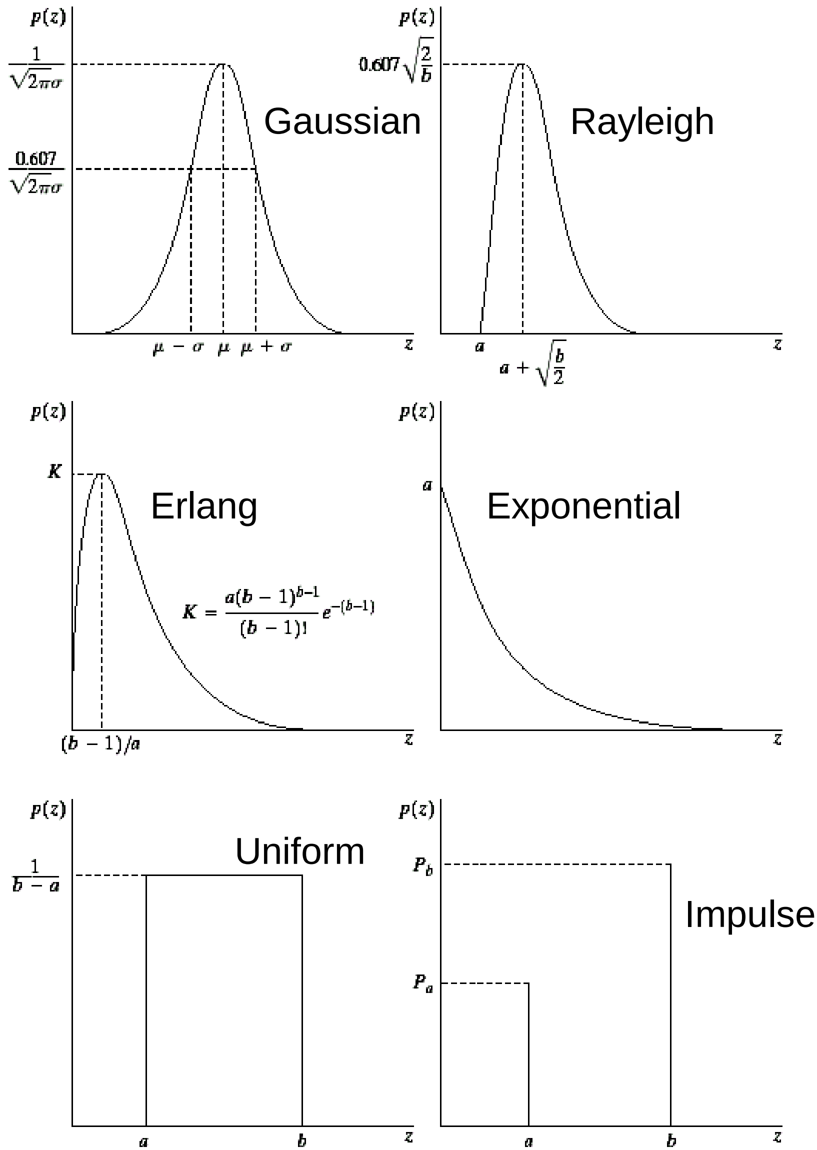



# Noise Models



There are many different models for the image noise term $\eta(x, y)$:

- Gaussian
  - Most common model
- Rayleigh
- Erlang
- Exponential
- Uniform
- Impulse
  - *Salt and pepper* noise

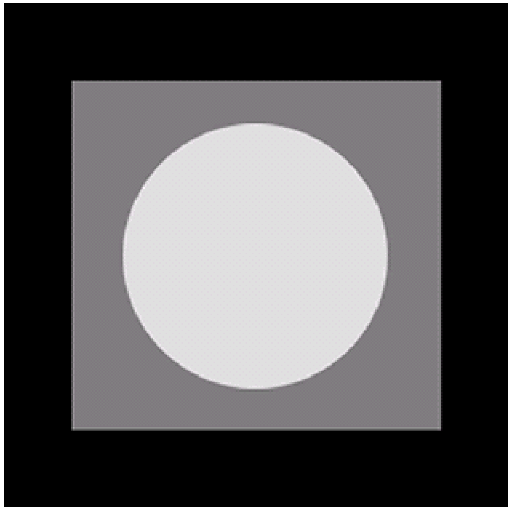


# Noise models

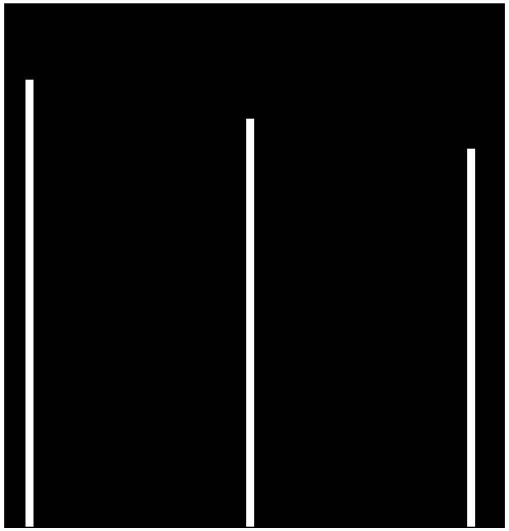
- Noise is characterized by its probability density function. Some of which are:
- Gaussian, Rayleigh, Uniform, Exponential, etc.
- By far Gaussian is the most popular model, because:
  - It is present widely in practice
  - Mathematical ease

# Noise Example

- The test pattern to the right is ideal for demonstrating the addition of noise
- The following slides will show the result of adding noise based on various models to this image

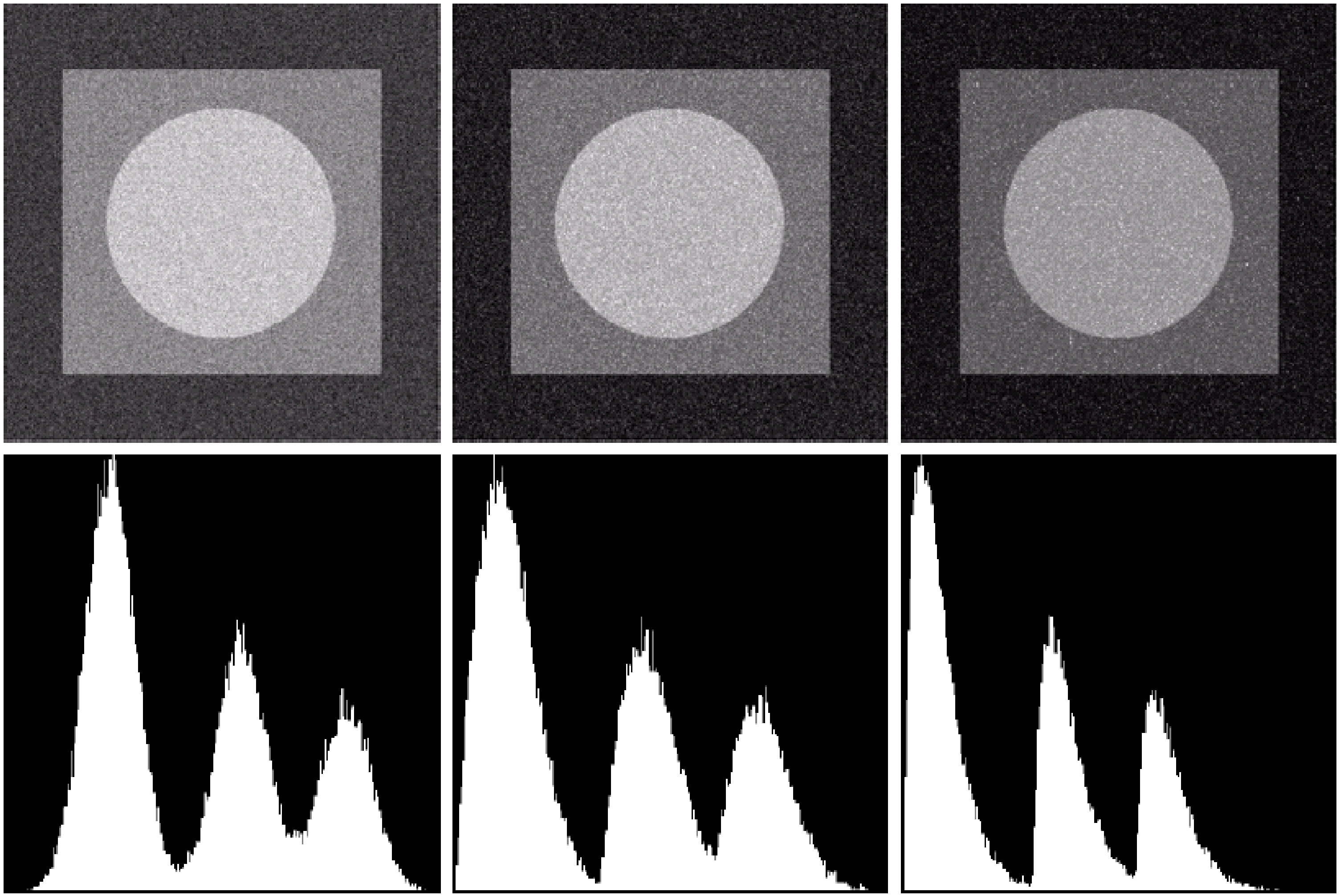


Image



Histogram

# Noise Example (cont…)



Gaussian Rayleigh Erlang

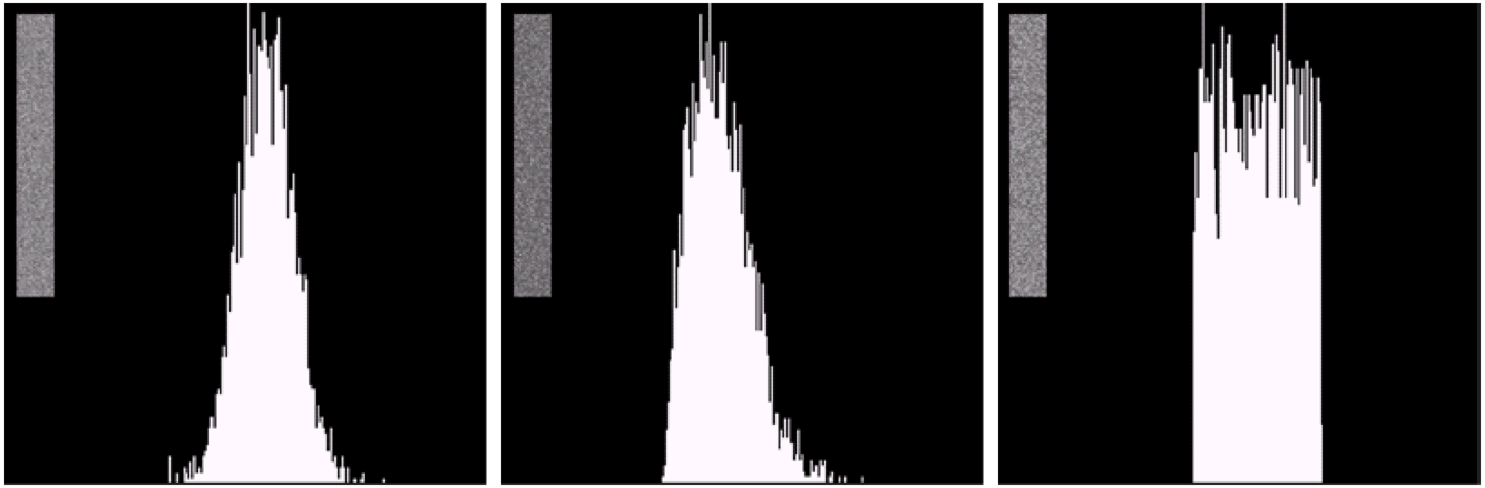
# Identifying noise model

- If the imaging system is available we can capture images of a uniform gray value object, example a black board uniformly illuminated.

- If images from the system are given, look for part of images which are uniform.

# Identifying noise

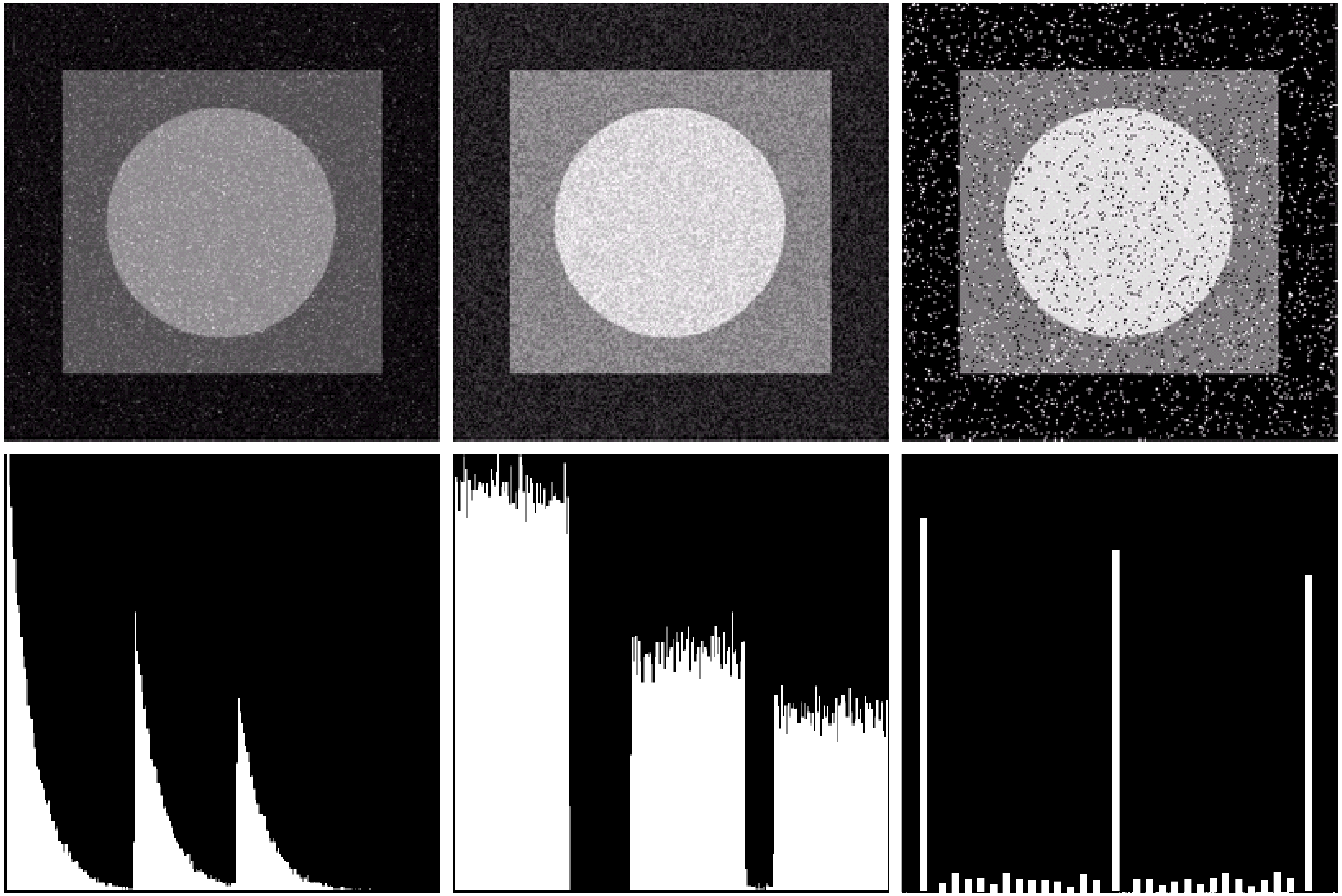
- Observing the histogram we can figure out which model the noise comes from.



a b c

**FIGURE 5.6** Histograms computed using small strips (shown as inserts) from (a) the Gaussian, (b) the Rayleigh, and (c) the uniform noisy images in Fig. 5.4.

# Noise Example (cont…)



# Estimation of noise parameters

- Noise is characterized by the pdf, which in turn depends on certain parameters.

- For example the Gaussian model depends only on the mean and variance and can easily be computed from the histogram.

$$\mu = \sum_{z_i \in S} z_i p(z_i)$$

$$\sigma^2 = \sum_{z_i \in S} (z_i - \mu)^2 p(z_i)$$

where p(zi) is the normalized histogram values.

# Restoration in presence of noise

- Degradation model: $g = f + n$

- Restoration = Enhancement
  - Mean filtering, Median filtering, Notch filters.

- Some new ones:
  - Geometric mean filter, Harmonic mean filters, Midpoint filter, etc.

# Filtering to Remove Noise

We can use spatial filters of different kinds to remove different kinds of noise

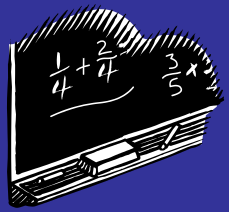
The *arithmetic mean* filter is a very simple one and is calculated as follows:

$$\hat{f}(x,y) = \frac{1}{mn} \sum_{(s,t)\in S_{xy}} g(s,t)$$

| | | |
|---|---|---|
| $^1/_9$ | $^1/_9$ | $^1/_9$ |
| $^1/_9$ | $^1/_9$ | $^1/_9$ |
| $^1/_9$ | $^1/_9$ | $^1/_9$ |

This is implemented as the simple smoothing filter

Blurs the image to remove noise



# Noise Removal Example

***Original Image*** *x*

| | | | | | | |
|---|---|---|---|---|---|---|
| 54 | 52 | 57 | 55 | 56 | 52 | 51 |
| 50 | 49 | 51 | 50 | 52 | 53 | 58 |
| 51 | 204 | 52 | 52 | 0 | 57 | 60 |
| 48 | 50 | 51 | 49 | 53 | 59 | 63 |
| 49 | 51 | 52 | 55 | 58 | 64 | 67 |
| 148 | 154 | 157 | 160 | 163 | 167 | 170 |
| 151 | 155 | 159 | 162 | 165 | 169 | 172 |

*y* ***Image f (x, y)***

***Filtered Image*** *x*

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

*y* ***Image f (x, y)***

# Other Means

There are different kinds of mean filters all of which exhibit slightly different behaviour:

– Geometric Mean

– Harmonic Mean

– Contraharmonic Mean

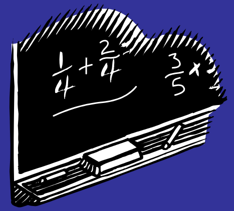
# Other Means (cont…)

There are other variants on the mean which can give different performance

**Geometric Mean:**

$$\hat{f}(x,y) = \left[ \prod_{(s,t)\in S_{xy}} g(s,t) \right]^{\frac{1}{mn}}$$

Achieves similar smoothing to the arithmetic mean, but tends to lose less image detail



# Noise Removal Example

***Original Image***

*x*

| | | | | | | |
|---|---|---|---|---|---|---|
| 54 | 52 | 57 | 55 | 56 | 52 | 51 |
| 50 | 49 | 51 | 50 | 52 | 53 | 58 |
| 51 | 204 | 52 | 52 | 0 | 57 | 60 |
| 48 | 50 | 51 | 49 | 53 | 59 | 63 |
| 49 | 51 | 52 | 55 | 58 | 64 | 67 |
| 148 | 154 | 157 | 160 | 163 | 167 | 170 |
| 151 | 155 | 159 | 162 | 165 | 169 | 172 |

***Image f (x, y)***

*y*

***Filtered Image***

*x*

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

***Image f (x, y)***

*y*

# Other Means (cont…)

**Harmonic Mean:**

$$\hat{f}(x,y) = \frac{mn}{\sum_{(s,t)\in S_{xy}} \frac{1}{g(s,t)}}$$

Works well for salt noise, but fails for pepper noise

Also does well for other kinds of noise such as Gaussian noise

# Noise Corruption Example

*Original Image*

*x*

| | | | | | | |
|---|---|---|---|---|---|---|
| 54 | 52 | 57 | 55 | 56 | 52 | 51 |
| 50 | 49 | 51 | 50 | 52 | 53 | 58 |
| 51 | 204 | 52 | 52 | 0 | 57 | 60 |
| 48 | 50 | 51 | 49 | 53 | 59 | 63 |
| 49 | 51 | 52 | 55 | 58 | 64 | 67 |
| 50 | 54 | 57 | 60 | 63 | 67 | 70 |
| 51 | 55 | 59 | 62 | 65 | 69 | 72 |

*Image f (x, y)*

*y*

*Filtered Image*

*x*

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

*Image f (x, y)*

*y*

# Other Means (cont…)

**Contraharmonic Mean:**

$$\hat{f}(x, y) = \frac{\sum_{(s,t)\in S_{xy}} g(s,t)^{Q+1}}{\sum_{(s,t)\in S_{xy}} g(s,t)^{Q}}$$

$Q$ is the *order* of the filter and adjusting its value changes the filter's behaviour

Positive values of $Q$ eliminate pepper noise

Negative values of $Q$ eliminate salt noise

# Noise Corruption Example

***Original Image***

*x*

| | | | | | | |
|---|---|---|---|---|---|---|
| 54 | 52 | 57 | 55 | 56 | 52 | 51 |
| 50 | 49 | 51 | 50 | 52 | 53 | 58 |
| 51 | 204 | 52 | 52 | 0 | 57 | 60 |
| 48 | 50 | 51 | 49 | 53 | 59 | 63 |
| 49 | 51 | 52 | 55 | 58 | 64 | 67 |
| 50 | 54 | 57 | 60 | 63 | 67 | 70 |
| 51 | 55 | 59 | 62 | 65 | 69 | 72 |

***Image f (x, y)***

*y*

***Filtered Image***

*x*

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

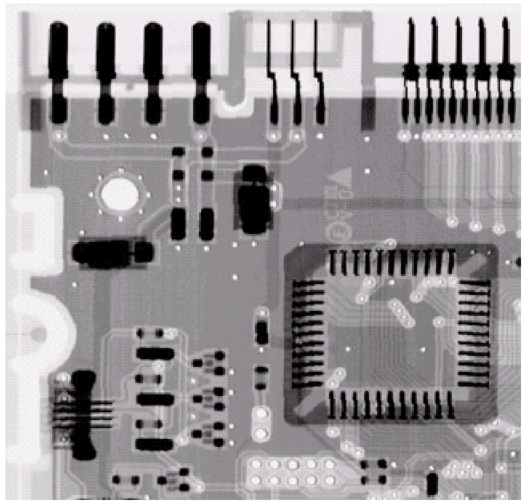
***Image f (x, y)***

*y*

# Noise Removal Examples

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Original
Image



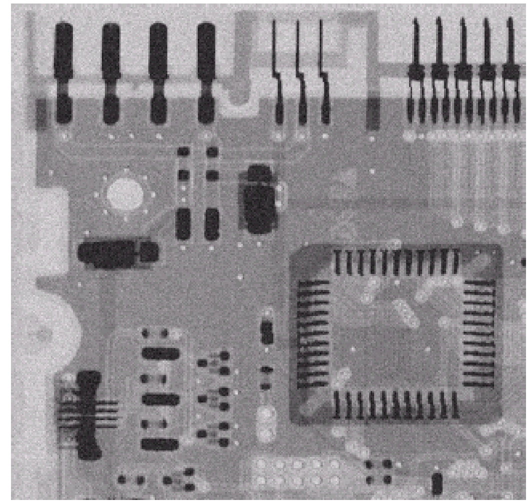
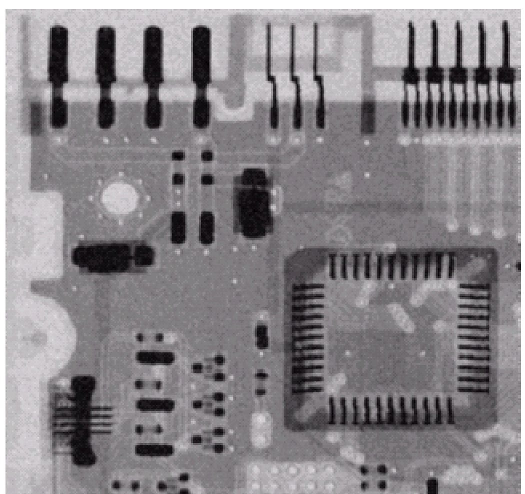
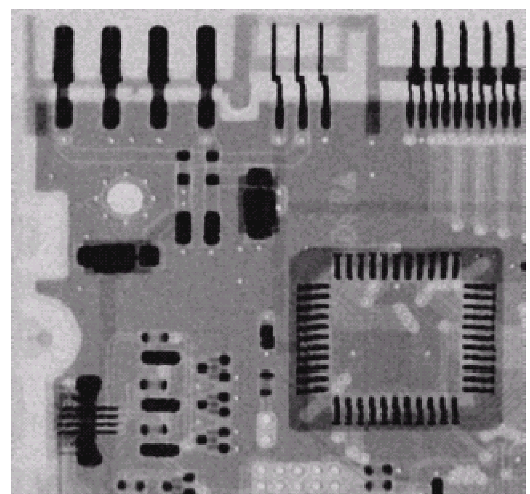


Image
Corrupted
By Gaussian
Noise

After A 3*3
Arithmetic
Mean Filter

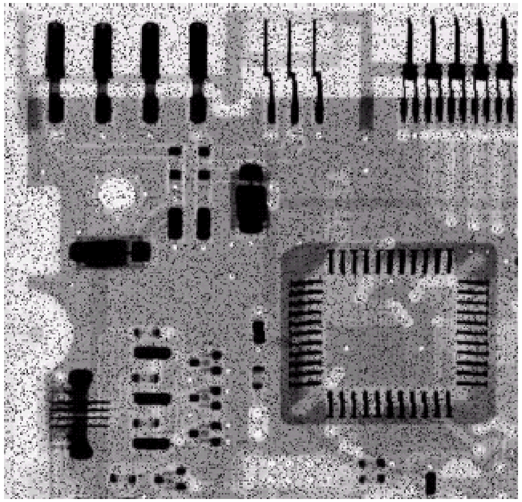




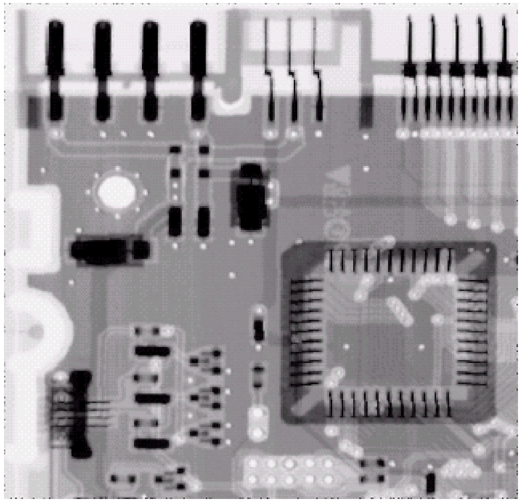
After A 3*3
Geometric
Mean Filter

# Noise Removal Examples (cont…)

Image Corrupted By Pepper Noise



Result of Filtering Above With 3*3 Contraharmonic Q=1.5



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

# Noise Removal Examples (cont…)



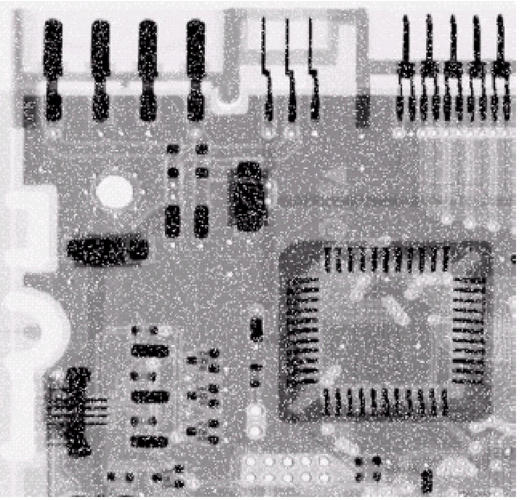
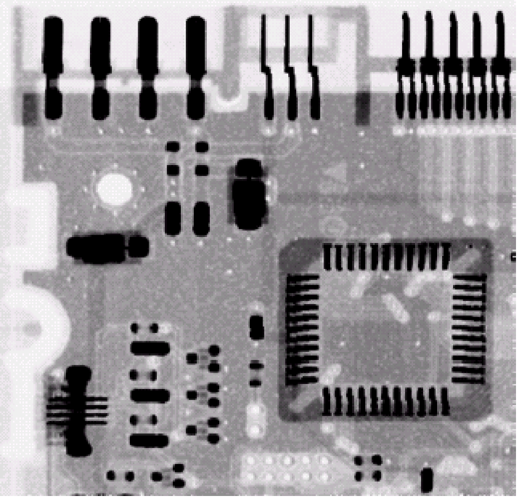


Image
Corrupted
By Salt
Noise

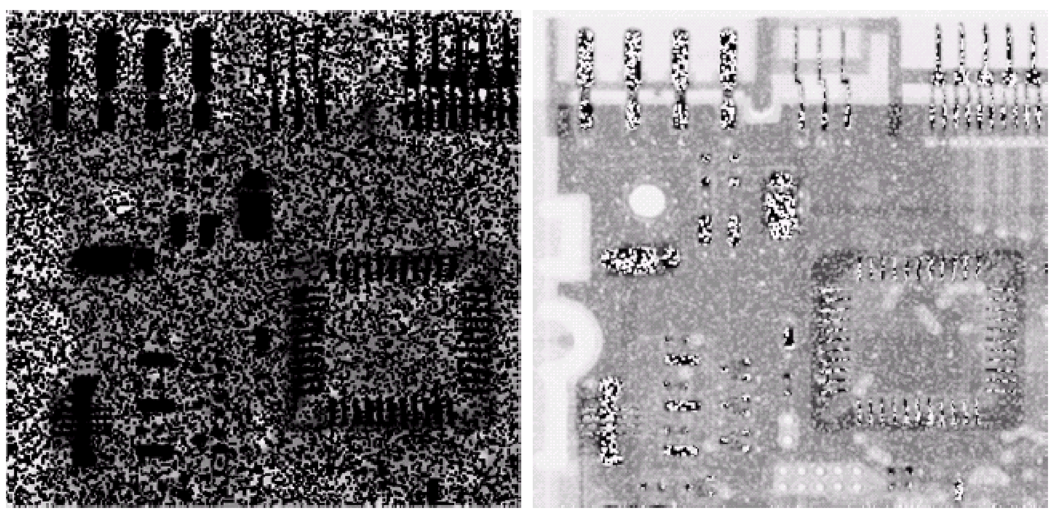


Result of
Filtering Above
With 3*3
Contraharmonic
Q=-1.5

# Contraharmonic Filter: Here Be Dragons



Choosing the wrong value for Q when using the contraharmonic filter can have drastic results



# Order Statistics Filters



Spatial filters that are based on ordering the pixel values that make up the neighbourhood operated on by the filter

Useful spatial filters include

- Median filter
- Max and min filter
- Midpoint filter
- Alpha trimmed mean filter

# Median Filter

**Median Filter:**

$$\hat{f}(x, y) = \underset{(s,t)\in S_{xy}}{median}\{g(s,t)\}$$

Excellent at noise removal, without the smoothing effects that can occur with other smoothing filters

Particularly good when salt and pepper noise is present

# Noise Corruption Example

***Original Image*** ***x***

| | | | | | | |
|---|---|---|---|---|---|---|
| 54 | 52 | 57 | 55 | 56 | 52 | 51 |
| 50 | 49 | 51 | 50 | 52 | 53 | 58 |
| 51 | 204 | 52 | 52 | 0 | 57 | 60 |
| 48 | 50 | 51 | 49 | 53 | 59 | 63 |
| 49 | 51 | 52 | 55 | 58 | 64 | 67 |
| 50 | 54 | 57 | 60 | 63 | 67 | 70 |
| 51 | 55 | 59 | 62 | 65 | 69 | 72 |

***Image f (x, y)***

***y***

***Filtered Image*** ***x***

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

***Image f (x, y)***

***y***

# Max and Min Filter

**Max Filter:**

$$\hat{f}(x, y) = \max_{(s,t) \in S_{xy}} \{g(s,t)\}$$

**Min Filter:**

$$\hat{f}(x, y) = \min_{(s,t) \in S_{xy}} \{g(s,t)\}$$

Max filter is good for pepper noise and min is good for salt noise

# Noise Corruption Example

***Original Image*** *x*

| | | | | | | |
|---|---|---|---|---|---|---|
| 54 | 52 | 57 | 55 | 56 | 52 | 51 |
| 50 | 49 | 51 | 50 | 52 | 53 | 58 |
| 51 | 204 | 52 | 52 | 0 | 57 | 60 |
| 48 | 50 | 51 | 49 | 53 | 59 | 63 |
| 49 | 51 | 52 | 55 | 58 | 64 | 67 |
| 50 | 54 | 57 | 60 | 63 | 67 | 70 |
| 51 | 55 | 59 | 62 | 65 | 69 | 72 |

***Image f (x, y)***

*y*

***Filtered Image*** *x*

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

***Image f (x, y)***

*y*

# Midpoint Filter

**Midpoint Filter:**

$$\hat{f}(x, y) = \frac{1}{2}\left[\max_{(s,t)\in S_{xy}}\{g(s,t)\} + \min_{(s,t)\in S_{xy}}\{g(s,t)\}\right]$$

Good for random Gaussian and uniform noise

# Noise Corruption Example

***Original Image*** ***x***

| | | | | | | |
|---|---|---|---|---|---|---|
| 54 | 52 | 57 | 55 | 56 | 52 | 51 |
| 50 | 49 | 51 | 50 | 52 | 53 | 58 |
| 51 | 204 | 52 | 52 | 0 | 57 | 60 |
| 48 | 50 | 51 | 49 | 53 | 59 | 63 |
| 49 | 51 | 52 | 55 | 58 | 64 | 67 |
| 50 | 54 | 57 | 60 | 63 | 67 | 70 |
| 51 | 55 | 59 | 62 | 65 | 69 | 72 |

***y***

***Image f (x, y)***

***Filtered Image*** ***x***

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

***y***

***Image f (x, y)***

# Alpha-Trimmed Mean Filter

**Alpha-Trimmed Mean Filter:**

$$\hat{f}(x, y) = \frac{1}{mn - d} \sum_{(s,t) \in S_{xy}} g_r(s, t)$$

We can delete the *d/2* lowest and *d/2* highest grey levels

So $g_r(s, t)$ represents the remaining $mn - d$ pixels

# Noise Corruption Example

*Original Image*

x

| | | | | | | |
|---|---|---|---|---|---|---|
| 54 | 52 | 57 | 55 | 56 | 52 | 51 |
| 50 | 49 | 51 | 50 | 52 | 53 | 58 |
| 51 | 204 | 52 | 52 | 0 | 57 | 60 |
| 48 | 50 | 51 | 49 | 53 | 59 | 63 |
| 49 | 51 | 52 | 55 | 58 | 64 | 67 |
| 50 | 54 | 57 | 60 | 63 | 67 | 70 |
| 51 | 55 | 59 | 62 | 65 | 69 | 72 |

y

*Image f (x, y)*

*Filtered Image*

x

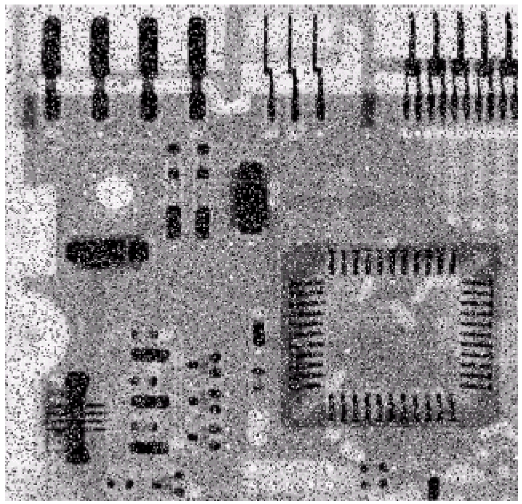
| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

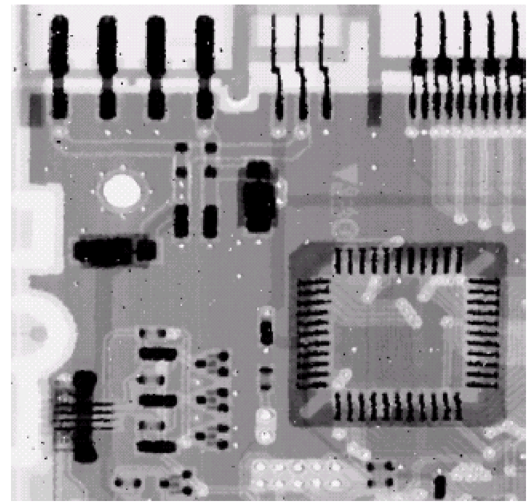
y

*Image f (x, y)*

# Noise Removal Examples

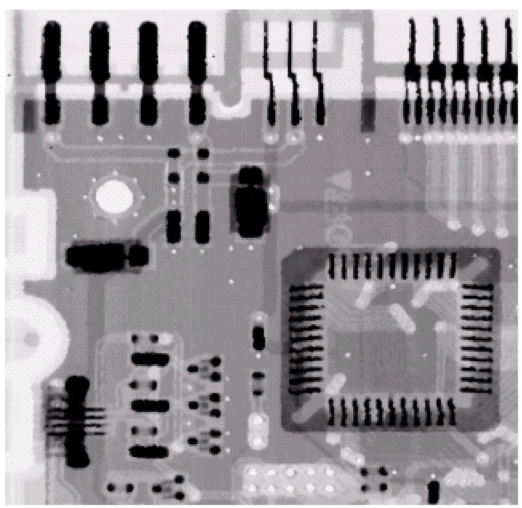
Image Corrupted By Salt And Pepper Noise

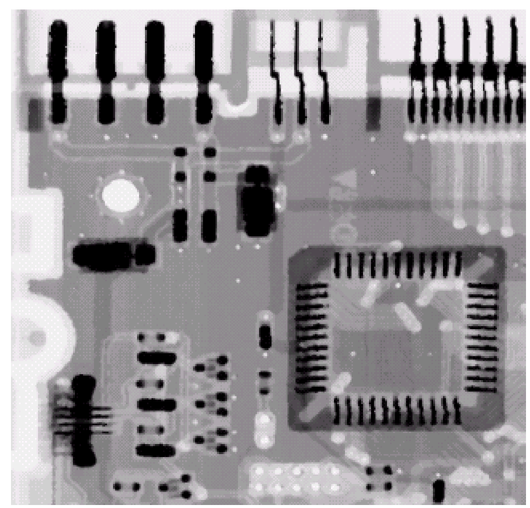




Result of 1 Pass With A 3*3 Median Filter

Result of 2 Passes With A 3*3 Median Filter

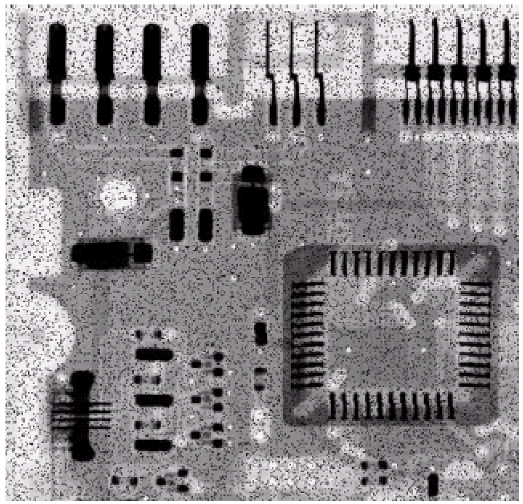




Result of 3 Passes With A 3*3 Median Filter

# Noise Removal Examples (cont…)

Image Corrupted By Pepper Noise



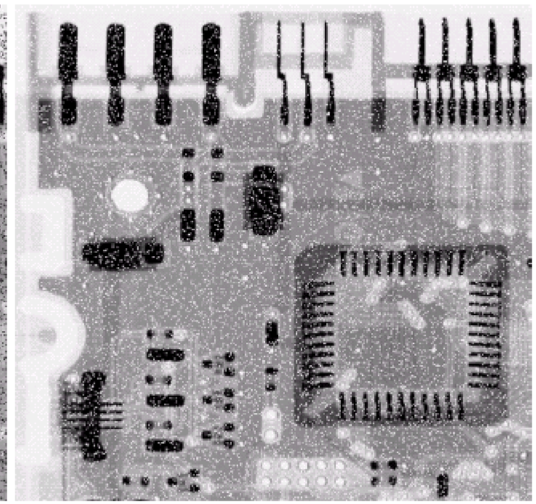
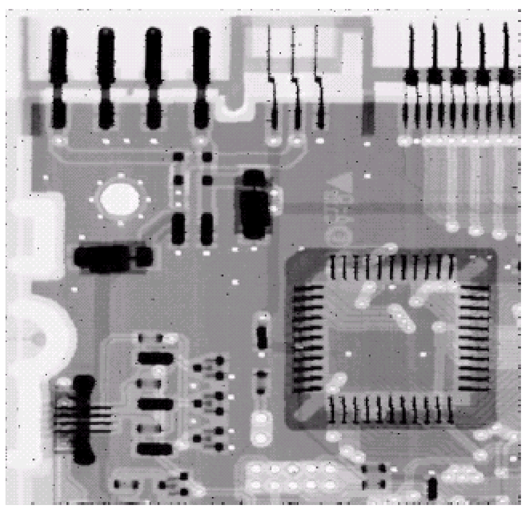
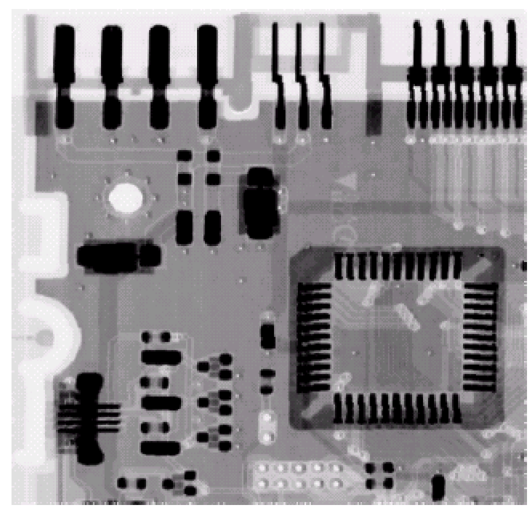


Image Corrupted By Salt Noise

Result Of Filtering Above With A 3*3 Max Filter





Result Of Filtering Above With A 3*3 Min Filter

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

# Noise Removal Examples (cont…)

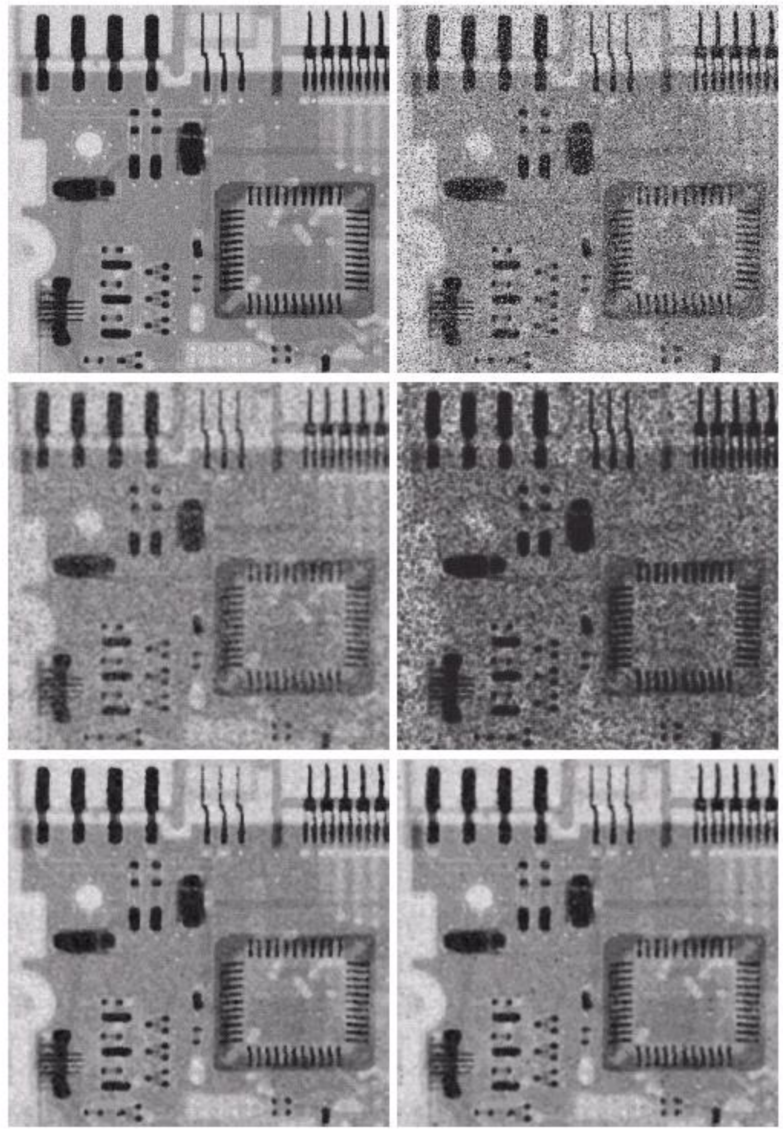


Image Corrupted By Uniform Noise

Image Further Corrupted By Salt and Pepper Noise

Filtered By 5*5 Arithmetic Mean Filter

Filtered By 5*5 Geometric Mean Filter

Filtered By 5*5 Median Filter

Filtered By 5*5 Alpha-Trimmed Mean Filter

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

# Adaptive Filters

- The filters discussed so far are applied to an entire image without any regard for how image characteristics vary from one point to another

- The behaviour of **adaptive filters** changes depending on the characteristics of the image inside the filter region

- We will take a look at the **adaptive median filter**

# Adaptive filters for handling noise

- Adapts to some statistical measures on a local neighborhood, usually mean and variance:
  - g(i,j): the noisy image gray-level value.
  - $\sigma_n^2$ : noise variance in the image.
  - $m_L$ : local mean in neighborhood.
  - $\sigma_L^2$ : local variance in neighborhood.

- $\sigma_n^2$ could be estimated from an uniform area in the given image.

# Adaptive local noise reduction filter

- If noise variance is zero ➔ Indicates no noise ➔ return the observed image.

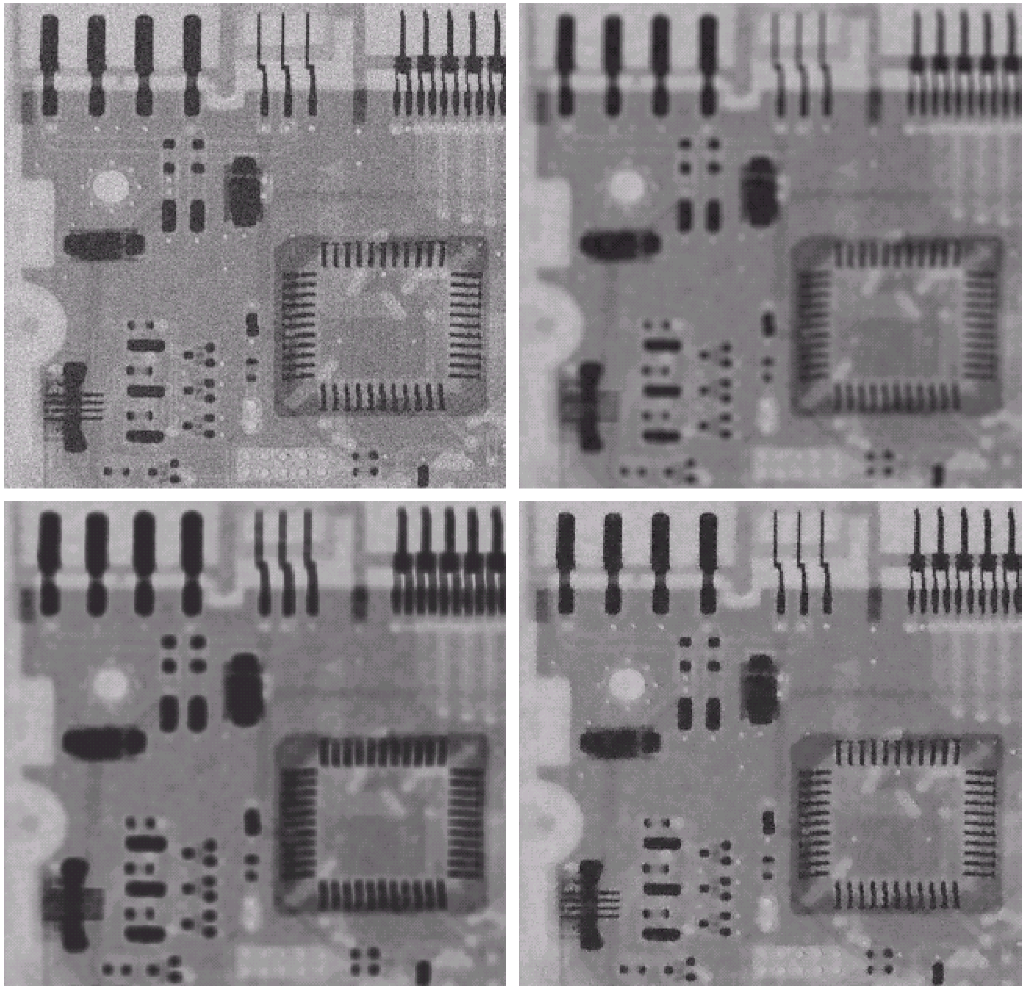
- If local variance is high compared to the noise variance ➔ presence of edge or a sharp feature➔ return a value close to observed grey-value.

- If two variances are equal➔ presence of noise➔ return the arithmetic mean

# Adaptive local noise reduction filter

- Filter equation:
  - $$\hat{f}(x,y) = g(x,y) - \frac{\sigma_\eta^2}{\sigma_L^2}[g(x,y) - m_L]$$
- How to handle $\sigma_\eta^2 > \sigma_L^2$, what does it mean?
- When $\sigma_\eta^2 > \sigma_L^2$, let $\frac{\sigma_\eta^2}{\sigma_L^2} = 1$ in the filter equation.

# Adaptive filter output



a b
c d

**FIGURE 5.13** (a) Image corrupted by additive Gaussian noise of zero mean and variance 1000. (b) Result of arithmetic mean filtering. (c) Result of geometric mean filtering. (d) Result of adaptive noise reduction filtering. All filters were of size $7 \times 7$.

# Adaptive Median Filtering

- The median filter performs relatively well on impulse noise as long as the spatial density of the impulse noise is not large

- The adaptive median filter can handle much more spatially dense impulse noise, and also performs some smoothing for non-impulse noise

- The key insight in the adaptive median filter is that the filter size changes depending on the characteristics of the image

# Adaptive Median Filtering (cont…)

- Remember that filtering looks at each original pixel image in turn and generates a new filtered pixel
- First examine the following notation:
  - $z_{min}$ = minimum grey level in $S_{xy}$
  - $z_{max}$ = maximum grey level in $S_{xy}$
  - $z_{med}$ = median of grey levels in $S_{xy}$
  - $z_{xy}$ = grey level at coordinates $(x, y)$
  - $S_{max}$ =maximum allowed size of $S_{xy}$

# Adaptive Median Filtering (cont…)

Level A: $A1 = z_{med} - z_{min}$

$A2 = z_{med} - z_{max}$

If $A1 > 0$ and $A2 < 0$, Go to level B

Else increase the window size

If window size $\leq S_{max}$ repeat level A

Else output $z_{xy}$

Level B: $B1 = z_{xy} - z_{min}$

$B2 = z_{xy} - z_{max}$

If $B1 > 0$ and $B2 < 0$, output $z_{xy}$

Else output $z_{med}$

# Adaptive Median Filtering (cont…)

The key to understanding the algorithm is to remember that the adaptive median filter has three purposes:

- – Remove impulse noise
- – Provide smoothing of other noise
- – Reduce distortion

# Adaptive Filtering Example



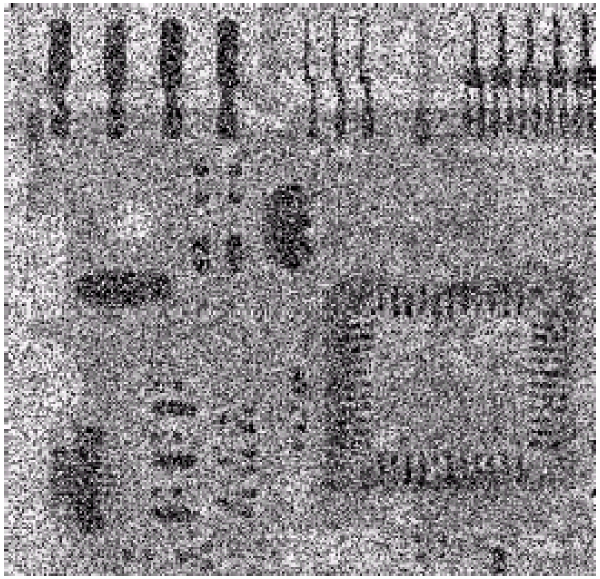
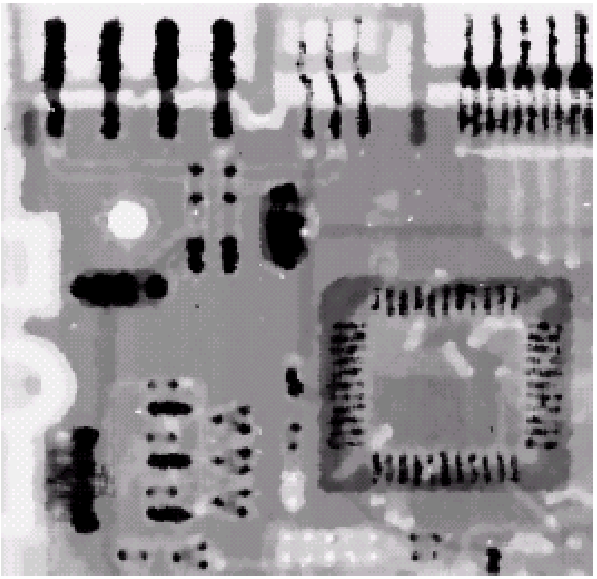
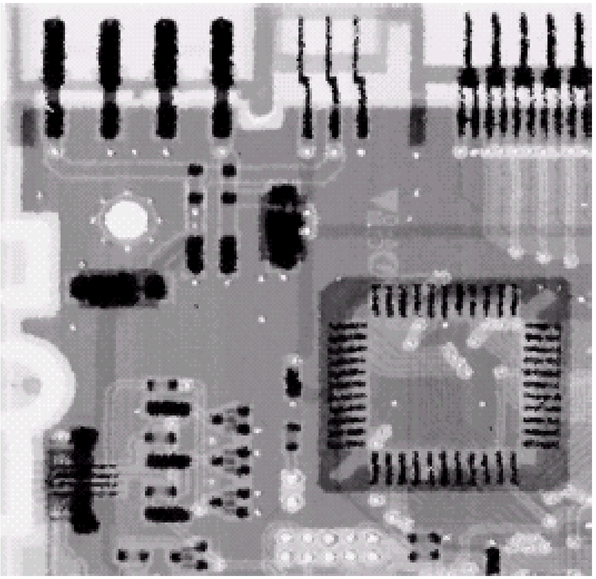


Image corrupted by salt and pepper noise with probabilities $P_a = P_b = 0.25$



Result of filtering with a 7 * 7 median filter

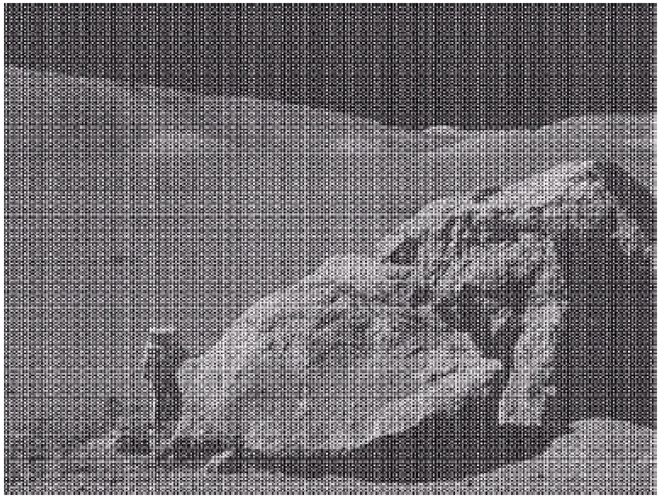


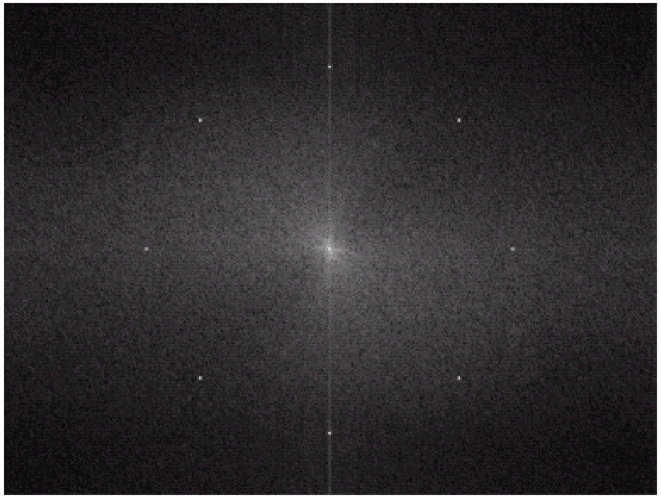
Result of adaptive median filtering with i = 7

# Periodic Noise



- It arises due to electrical or electromagnetic interference
- Gives rise to regular noise patterns in an image
- Frequency domain techniques in the Fourier domain are most effective at removing periodic noise





# Band Reject Filters

- Removing periodic noise form an image involves removing a particular range of frequencies from that image
- *The reject* filters can be used for this purpose

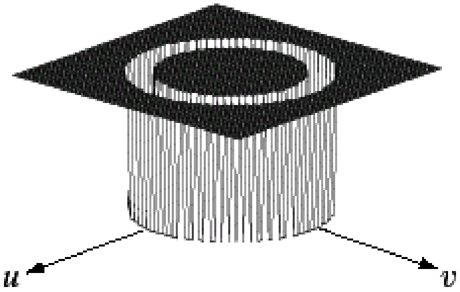
An ideal band reject filter is given as follows:

$$H(u,v) = \begin{cases} 1 & if \;\; D(u,v) < D_0 - \dfrac{W}{2} \\ 0 & if \;\; D_0 - \dfrac{W}{2} \le D(u,v) \le D_0 + \dfrac{W}{2} \\ 1 & if \;\; D(u,v) > D_0 + \dfrac{W}{2} \end{cases}$$

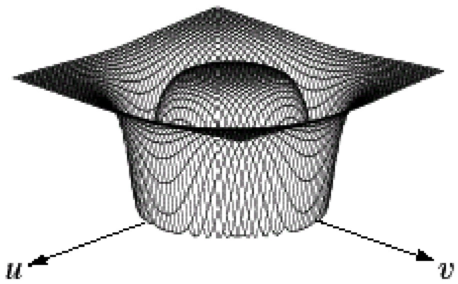
# Band Reject Filters (cont…)



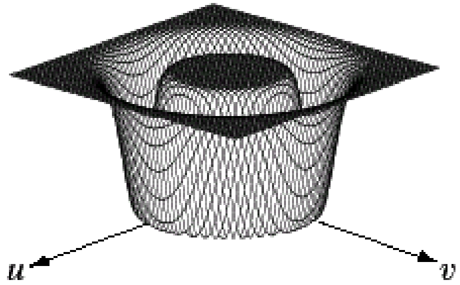
The ideal band reject filter is shown below, along with Butterworth and Gaussian versions of the filter




Ideal Band
Reject Filter




Butterworth
Band Reject
Filter (of order 1)

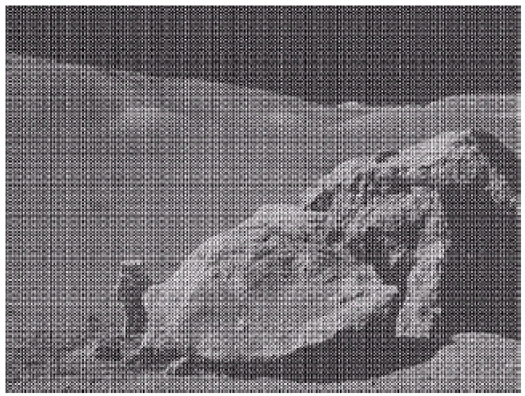



Gaussian
Band Reject
Filter

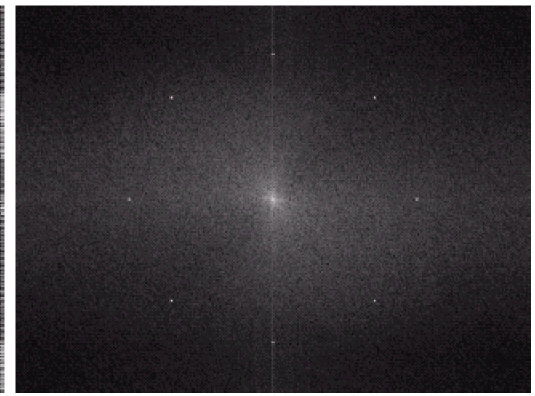
# Band Reject Filter Example

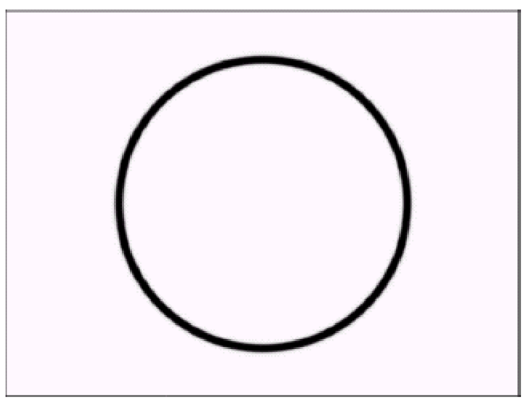


Image corrupted by sinusoidal noise



Fourier spectrum of corrupted image





Butterworth band reject filter

Filtered image

# Summary

- In this lecture we will look at image restoration for noise removal
- Restoration is slightly more objective than enhancement
- Spatial domain techniques are particularly useful for removing random noise
- Frequency domain techniques are particularly useful for removing periodic noise